Certification of MPC-based zonal controller security properties using accuracy-aware machine learning proxies

Pierre HOUDOUIN, Manuel RUIZ, Lucas SALUDJIAN, Patrick PANCIATICI
French Transmission System Operator, RTE, Paris, France
{pierre.houdouin, manuel.ruiz, patrick.panciatici}@rte-france.com

***Abstract*—The fast growth of renewable energies increases the power congestion risk. To address this issue, the French Transmission System Operator (RTE) has developed closed-loop controllers to handle congestion. To guarantee their proper functioning, RTE wishes to estimate the probability that the controllers ensure the equipment's safety. The naive approach to estimating this probability relies on simulating many randomly drawn scenarios and then use all the outcomes to build a confidence interval around the probability. Although theory ensures convergence, the computational cost of power system simulations makes such a process intractable.**

The present paper aims to propose a faster process using machine-learning-based proxies. The amount of required simulations is significantly reduced thanks to an accuracy-aware proxy built with Multivariate Gaussian Processes. However, using a proxy instead of the simulator adds uncertainty to the outcomes. An adaptation of the Central Limit Theorem is thus proposed to include the uncertainty of the outcomes predicted with the proxy into the confidence interval. As a case study, we designed a simple simulator that is tested on a small network. Results show that the proxy learns to approximate the simulator's answer accurately, allowing a significant time gain for the machine-learning-based process.

***Index Terms*—Certification of security properties, Congestion management, Multivariate Gaussian processes, NAZA, Proxies**

I. Introduction

Integrating renewable energies on a large scale poses challenges in the operation and management of power systems. It leads to unpredictable and variable flow injections into transmission lines, increasing the risk of power congestion. To address these challenges, the French Transmission System Operator (TSO), RTE, has adopted a decentralized management approach, dividing the entire system into sub-transmission areas (zones). Real-time constraints within each zone are managed through a local closed-loop controller called NAZA [1], [2], [3]. Designed to handle local problems with local actions, it enables the management of battery devices, topological modifications, and curtailment of renewable production [4] inside the zone. Alongside the massive deployment of these controllers across the network, RTE aims to obtain guarantees regarding their proper functioning. Given a set of renewable power production scenarios, RTE seeks to compute the probability $\mathbf{p_{safe}}$ of NAZA, ensuring the equipment's safety. Equipment's safety is ensured if congestions are avoided during the scenario simulation. If any line in the zone becomes overloaded during the simulation, it is considered a security threat.

The estimation of $\mathbf{p_{safe}}$ relies on the law of large numbers. The most basic approach is the brute-force process: scenarios are randomly drawn and simulated until $\mathbf{p_{safe}}$ is accurately estimated. Central Limit Theorem (CLT) [5] ensures the convergence of this process. The more iterations are performed, the more accurate the estimation becomes [6]. In our case, only extreme and thus rare-to-observe scenarios will likely pose a security threat. A large number of scenarios must, therefore, be drawn to observe enough threatening situations and obtain a reliable estimation of the probability. Although it theoretically works, the drawback of the brute-force process is that it requires a lot of simulations. As power system simulations are computationally costly, such a process is intractable in this case. However, the outcome of many scenarios can be figured out without a simulation. For instance, a scenario without wind will surely not lead to congestion. Hence, learning a proxy of the simulator to avoid unnecessary simulations and focus only on the interesting ones can lead to considerable time gain.

The use of proxies to reduce global computational costs meets a growing interest in all domains of power systems. In 2016, Canyasse et al. [7] used supervised learning algorithms to build real-time proxies for solving ACOPF. Two years later, Duchesne [8] investigated machine-learning proxies to deal with SCOPF in the context of operation planning. Also, in 2018, Dalal et al. [9], [10] considered the Nearest Neighbors algorithm to predict short-term decision outcomes and applied it to the outage scheduling problem. More recently, in 2022, Chen et al. [11] proposed a deep-learning-based proxy to solve the SCED problem and handle real-time applications efficiently.

In this paper, we will also use a proxy to address the computational limitations of the brute-force approach. Our new proxy-based process aims to avoid useless simulations and achieve a faster estimation of $\mathbf{p_{safe}}$. Using batches of simulations, we train a **multivariate Gaussian process (MGP)** [12] to predict the scenario outcome. **Gaussian process (GP)**

Submitted to the 23nd Power Systems Computation Conference (PSCC 2024).

also became very popular in the power systems community [13], [14], [15]. Indeed, they provide exact confidence intervals around the prediction and enable the incorporation of prior knowledge of the system into the proxy [16]. Parameters of the conditional distribution are continuously updated with the new batch of data to enhance its accuracy throughout the process. The confidence interval provided with the GP's prediction enables permanent assessment of the prediction's quality. If the uncertainty in the prediction of the scenario outcome is acceptable, we keep the prediction and avoid a simulation. Otherwise, we perform a simulation. This leads to significantly sped-up iterations when the simulation is not performed. Finally, the CLT is adapted using the Lyapunov version to include the uncertainty of the proxy's predictions in the confidence interval of $\mathbf{p_{safe}}$'s estimation.

The objective of this work is two-fold:

- Train an accurate proxy of a black-box simulator using a MGP
- Show that with an accurate proxy, the machine-learning-based process converges much faster toward $\mathbf{p_{safe}}$ for a given precision, with even a greater gain for demanding precision

The article's structure is as follows: Section II introduces the notations, reminders on the MGP, and problem formulation. Section III presents both the brute-force and the proxy-base processes. Section IV provides the computational results of the case study. Conclusions and future perspectives are discussed in Section V.

II. Preliminaries and problem formulation

A. Notations

Throughout this paper, upper-case (lower-case) boldface letters will be used for matrices (column vectors), and $(.)^T$ denotes the transposition. Given a zone of the electric network, we use the following notations :

- $\mathcal{Z}_L = \{L_1, ..., L_L\}$ is the set of lines in the considered zone, L is its cardinality
- $\mathcal{Z}_N = \{N_1, ..., N_N\}$ is the set of nodes in the considered zone, N is its cardinality
- P_n is the allowed renewable power injection at node N_n
- PA_n is the available renewable power injection at node N_n
- P_n^{max} is the maximum renewable power that can be produced at node N_n, and $\frac{PA_n}{P_n^{max}} \in [0,1]$ is the relative available renewable power injection
- F_l and $\bar{F}_l$ are respectively the power flow and the IST of line L_l

B. Multivariate gaussian process

Consider an unknown function $h : \mathbb{R}^N \to \mathbb{R}^L$ that represents the non-deterministic answer $\boldsymbol{y} \in \mathbb{R}^L$ of a system to an input $\boldsymbol{x} \in \mathbb{R}^N$. We model $\boldsymbol{y} = h(\boldsymbol{x}) + \mathcal{N}(0, \sigma_0^2 \boldsymbol{I})$. Multivariate Gaussian process [17] (MGP) regression aims to learn the underlying dynamic of h by supposing that it is the realization of a multivariate stochastic Gaussian process. The following elements fully characterize the multivariate stochastic process:

- Its mean function : $\mu : \mathbb{R}^N \to \mathbb{R}^L$
- Its correlation (kernel) function : $k : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}$
- Its output covariance matrix : $\boldsymbol{\Omega}$, a $L \times L$ matrix

Prior knowledge of the system's dynamic can be incorporated into these three elements. For a given $\boldsymbol{x} \in \mathbb{R}^N$, $\mu(\boldsymbol{x})$ represents the prior expected value of $h(\boldsymbol{x})$. Usually, as no specific information about h is known, the choice is $\mu = 0$, which is also our choice here. The kernel function $k(\boldsymbol{x_1}, \boldsymbol{x_2})$ measures to what extent $\boldsymbol{y_1}$ and $\boldsymbol{y_2}$, the outputs of $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$, are correlated. Its selection has to be adapted to the problem. For our problem, we choose the stationary squared exponential kernel function :

$$k(\boldsymbol{x_1}, \boldsymbol{x_2}) = \sigma_f^2 \exp\left(-\frac{|\boldsymbol{x_1} - \boldsymbol{x_2}|^2}{2l^2}\right) + \sigma_0^2 \delta(\boldsymbol{x_1}, \boldsymbol{x_2})$$

It is a parametric function with parameters $\boldsymbol{\theta} = \left[\sigma_0^2, \sigma_f^2, l\right]$. Parameter σ_0^2 represents the variance of the system's answer to the same input, σ_f^2 reflects the signal variance, and l corresponds to the characteristic length-scale of the kernel. The optimal parameters are learned by maximizing the likelihood [12], [18]. The squared exponential kernel results in a smooth prior on h and reflects that similar scenarios are expected to have similar threat potential for the network. Knowing one scenario's outcome only provides local information about neighbors' scenarios' outcomes. The correlation between their output indeed decreases exponentially with the distance. Finally, the process's output covariance matrix reflects the correlations between the output's coordinates. This matrix allows the posterior distribution of the MGP to preserve the covariance structure of the output. The choice of $\boldsymbol{\Omega} = I_L$ implies that coordinates are decorrelated. The MGP then boils down to training L separate univariate GP to predict each coordinate independently. In this work, $\boldsymbol{\Omega}$ is fixed at the beginning of the process. It represents the line's maximum flow correlation and is chosen appropriately regarding the network's structure. An extension of this work to integrate an estimation of $\boldsymbol{\Omega}$ across the process's iterations is planned. We now define the following variables :

- $\boldsymbol{X_m} = [\boldsymbol{x_1}, ..., \boldsymbol{x_m}]$ the inputs vectors
- $\boldsymbol{Y_m} = \left[\boldsymbol{y_1}^T, ..., \boldsymbol{y_m}^T\right]$ the corresponding simulated outputs
- $\boldsymbol{M_m} = [\mu(\boldsymbol{x_1}), ..., \mu(\boldsymbol{x_m})]$ the corresponding mean vectors
- $\boldsymbol{\Sigma_m} = (k(\boldsymbol{x_i}, \boldsymbol{x_j}))_{i,j \in [1,m]}$ the covariance matrix between the input vectors
- $\boldsymbol{x_{m+1}}$ a new input vector
- $\boldsymbol{\Sigma_{[1,m],m+1}} = [k(\boldsymbol{x_1}, \boldsymbol{x_{m+1}}), ..., k(\boldsymbol{x_m}, \boldsymbol{x_{m+1}})]$ the covariance vector between the new input and the old ones

The joint distribution of $\boldsymbol{Y_{m+1}}$ is a matrix-variate gaussian distribution [19] $\mathcal{MN}\left(\boldsymbol{M_{m+1}}, \boldsymbol{\Sigma_{m+1}}, \boldsymbol{\Omega}\right)$ [17]. Similarly to the univariate GP, we can derive the conditional distribution of $\boldsymbol{y_{m+1}} | \boldsymbol{y_1}, ..., \boldsymbol{y_m}$:

$$\boldsymbol{\mu_*} = \mu(\boldsymbol{x_{m+1}}) + \boldsymbol{\Sigma_{[1,m],m+1}}^T \boldsymbol{\Sigma_m}^{-1}(\boldsymbol{Y_m} - \boldsymbol{M_m})$$
$$\sigma_* = k\left(\boldsymbol{x_{m+1}}, \boldsymbol{x_{m+1}}\right) - \boldsymbol{\Sigma_{[1,m],m+1}}^T \boldsymbol{\Sigma_m}^{-1} \boldsymbol{\Sigma_{[1,m],m+1}}^T$$
$$\boldsymbol{y_{m+1}} \in \mathbb{R}^L \sim \mathcal{N}\left(\boldsymbol{\mu_*}, \sigma_* \boldsymbol{\Omega}\right)$$

The obtained multivariate conditional distribution is very similar to the univariate case. The observed points matter for the mean $\boldsymbol{\mu_*}$ and the covariance matrix's scale σ_*. The covariance matrix's shape is always $\boldsymbol{\Omega}$, which ensures that the conditional distribution preserves the correlations between the output coordinates.

C. Problem formulation

Given :

- A zone of the power network
- A black-box simulator $\mathcal{S}$ to simulate the zone's behavior in answer to power injections
- A set of renewable power injection scenarios $\mathcal{X}$

, we aim to estimate the probability $\mathbf{p_{safe}}$ to draw a scenario $\boldsymbol{x} \in \mathcal{X}$ well-handled by NAZA, i.e., that do not lead to overloaded lines.

Hypothesis on the zone

The zone is composed of L lines, N generator nodes, and a NAZA controller to handle congestion management. We consider the following working hypothesis :

1) NAZA controller can curtail each node's renewable energy power injections, thus reducing $P_n \leq PA_n$.
2) Each generator produces the maximum available or allowed power
3) The loads are constants
4) DC approximation: we neglect the reactive power

Renewable energy generation scenario

We model a renewable energy production scenario $\boldsymbol{x}$ by the vector of the N available renewable power injections at each network node. To facilitate the MGP learning, we draw the relative available renewable power injection. Our set of scenarios $\mathcal{X}$ then boils down to $[0,1]^N$, which is easier to explore and will be referred to as the *information structure*. To preserve the spatial correlation of the nodes, the power injection vector $\boldsymbol{x}$ is generated using a truncated multivariate Gaussian distribution.

$$\boldsymbol{x} = \begin{pmatrix} x_1 \\ . \\ . \\ . \\ x_N \end{pmatrix} \sim \mathcal{N}_{|\mathcal{X}}\left(0, \boldsymbol{\Sigma_{nodes}}\right) \longrightarrow \begin{pmatrix} PA_1 = x_1 \times P_1^{max} \\ . \\ . \\ . \\ PA_N = x_N \times P_N^{max} \end{pmatrix}$$

We consider here a very restrictive subspace of low-dimensional scenarios. Our main goal is to show how an MGP can learn the multivariate answer of a black-box function. Future work will deal with more complex scenarios, including a temporal aspect, to simulate the network's behavior on many time steps.

Network simulation

The network's evolution is simulated using a *network simulator* $\mathcal{S}$. It takes as input a renewable power production scenario $\boldsymbol{x}$ and returns the vector of the lines' relative power flow $\boldsymbol{y} \in \mathbb{R}^L$. For line l, the relative power flow is $\boldsymbol{y}^{(l)} = \frac{F_l}{\overline{F_l}}$. The *network simulator* used here is a simple simulator implemented by the authors (more details are provided in the experiments section). Again, an extension of this work, on a real zone of the French transmission network and using a real network simulator [20] is planned.

Well-handled scenarios

A scenario $\boldsymbol{x}$ is considered well-handled by NAZA if no congestions on any lines occur. Line L_l is considered congested if its power flow exceeds the IST. Thus, denoting by f the Boolean function that indicates whether a scenario is well-handled or represents a security threat, we have :

$$f\left(\boldsymbol{y}\right) = z = \begin{cases} 1 & \text{if } \forall l \in [1, L], \boldsymbol{y}^{(l)} <= 1 \\ 0 & \text{if } \exists l \in [1, L], \boldsymbol{y}^{(l)} > 1 \end{cases}$$

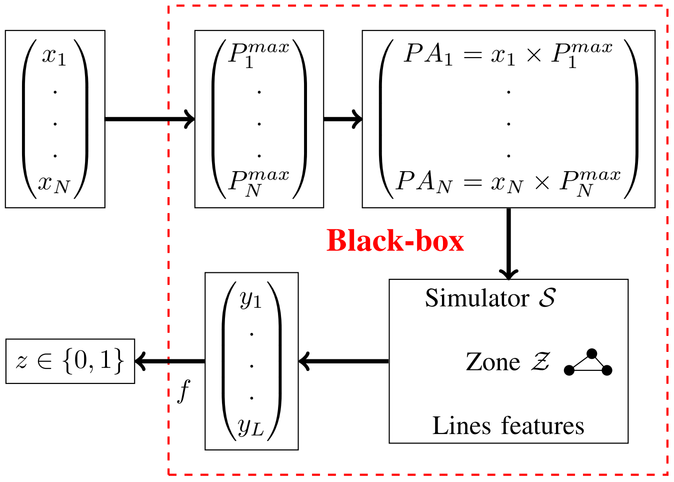


Fig. 1. Black-box function taking as input a renewable energy generation scenario $\boldsymbol{x}$ and returning 0 if a congestion occurred, 1 if security was ensured

The zone, the nodes, lines features, and the simulator constitute a *black-box function*, as illustrated in Fig.1. The goal of this work is to estimate as precisely and as fast as possible the probability for the black-box function to return 1: $\mathbf{p_{safe}} = \mathbb{P}\left(f \circ \mathcal{S}(\boldsymbol{x}) = 1\right)$. The following section will detail two processes to estimate $\mathbf{p_{safe}}$.

III. Processes presentation

A. Brute-force process

The brute-force process is the vanilla way to estimate $\mathbf{p_{safe}}$. We randomly sample scenarios in $\mathcal{X}$, compute $f \circ \mathcal{S}$ using the network simulator, and use the CLT to build a confidence interval around $\mathbf{p_{safe}}$. We iterate the sampling until a satisfying confidence interval is reached. Given the m simulation results $z_1, ..., z_m$, $\overline{z_m} = \frac{1}{m}\sum_{i=1}^{m} z_i$ their average, Q_α the $1 - \alpha$

Algorithm 1 Brute-force process

Inputs :

- scenario sampler $\mathbb{X}$
- network simulator $\mathcal{S}$
- confidence interval precise enough : $CI_{good}?$ tool
- confidence interval level $1-\alpha$

Output : $I = [p_{min}, p_{max}]$, $1-\alpha$ confidence interval level around $\mathbf{p_{safe}}$

```
1: Initialize $m = 0$ and $I_m = [0,1]$
2: Initialize the bag of outputs $\mathcal{B} = \{\}$
3: while not $CI_{good}?(I_m)$ do
4:     Draw a scenario $\boldsymbol{x_m}$ using $\mathbb{X}$
5:     Increase $m = m + 1$
6:     Compute the outcome $z_m = f \circ \mathcal{S}(\boldsymbol{x_m})$
7:     Add outcome $z_m$ to the bag $\mathcal{B} = \mathcal{B} \cup \{z_m\}$
8:     Compute $I_m$ of level $1-\alpha$ using $\mathcal{B}$
9: end while
```

quantile of the standard normal distribution, the asymptotic confidence interval [6] of level $1-\alpha$ around $\mathbf{p_{safe}}$ is :

$$I_\alpha = [p_{min}, p_{max}]$$

$$p_{min} = \frac{\overline{z_m} + \frac{Q^2_{\frac{\alpha}{2}}}{2m} - Q_{\frac{\alpha}{2}}\sqrt{\frac{\overline{z_m}(1-\overline{z_m})}{m} + \frac{q^2_{\frac{\alpha}{2}}}{4m^2}}}{1 + \frac{Q^2_{\frac{\alpha}{2}}}{m}}$$

$$p_{max} = \frac{\overline{z_m} + \frac{Q^2_{\frac{\alpha}{2}}}{2m} + Q_{\frac{\alpha}{2}}\sqrt{\frac{\overline{z_m}(1-\overline{z_m})}{m} + \frac{q^2_{\frac{\alpha}{2}}}{4m^2}}}{1 + \frac{Q^2_{\frac{\alpha}{2}}}{m}}$$

The brute-force process is summarized in Algorithm 1.

B. Proxy-based process

The issue with the brute-force process is the very high amount of required simulations to get an accurate estimation of $\mathbf{p_{safe}}$, especially if $\mathbf{p_{safe}}$ is close to 0 or 1. Fig.2 depicts the number of simulations required for the brute-force process to reach the desired relative precision on the estimated probability for different values of $\mathbf{p_{safe}}$. To reduce the number of required simulations, we propose to learn the response $\boldsymbol{y} = \mathcal{S}(\boldsymbol{x})$ of the simulator to a scenario. We train a MGP-based proxy $\tilde{\mathcal{S}}$ with the previously simulated scenarios.

In this new process, we sample batches of scenarios. For each input scenario $\boldsymbol{x_i}$, the MGP-based proxy provides the probability density function (PDF) g_i of the output $\boldsymbol{y_i} \in \mathbb{R}^L$. Integrating this PDF allows us to compute the probability $p_i = \int_{[1,\infty]^L} g_i(y)dy$ for scenario $\boldsymbol{x_i}$ to lead to congestion. If the confidence in the outcome is high enough, i.e., if p_i is either very small or very high, we keep the prediction and avoid a simulation. Otherwise, a simulation is performed to obtain the outcome. Fig.3 illustrates the decision mechanism to decide when to keep the prediction or simulate the outcome. At

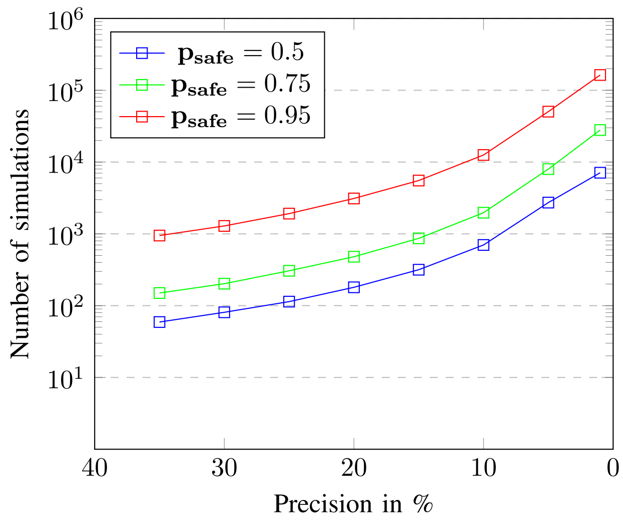
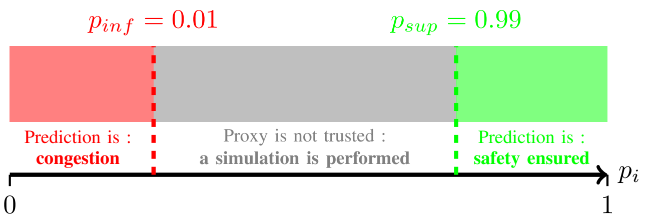


Fig. 2. Number of simulations required for the brute-force process to reach a desired relative precision, depending on $\mathbf{p_{safe}}$ probability, each point is the average of 100 simulations

the beginning of the process, only a few outcomes are available to train the proxy. Its quality is poor, and the prediction's confidence is often insufficient to avoid the simulation. Most scenarios are thus simulated. However, the more samples are drawn, the better the proxy becomes. At some point, most predictions will be confident enough to avoid the simulation, resulting in a considerable time gain for the probability estimation. The proxy-based process is detailed in Algorithm 2.



$p_{inf} = 0.01$
$p_{sup} = 0.99$
Prediction is : congestion
Proxy is not trusted : a simulation is performed
Prediction is : safety ensured
p_i
0
1

Fig. 3. Proxy use decision mechanism

We compute the confidence interval using only simulated outcomes in the brute-force process. There is no uncertainty on any z_i. In the machine learning process, it is different. When the outcome is predicted, z_i is no longer known with 100% confidence because $0 \le p_i \le 1$. Such uncertainty must be considered in the confidence interval around the estimation of $\mathbf{p_{safe}}$. The classic version of the CLT must be adapted to our problem.

C. Central Limit Theorem adaptation

Let $\boldsymbol{x_1}, ..., \boldsymbol{x_m} \in \mathcal{X}$ be m scenarios :

- $z_1, ..., z_m \in \{0,1\}$ denote the true answers given by the simulator
- $\forall i \in [1,m], \mathbb{P}(z_i = 1) = \mathbf{p_{safe}}$
- $w_1, ..., w_m \in \{0,1\}$ denote the answers' prediction
- We define $\forall i \in [1,p], q_i = \mathbb{P}(w_i = z_i)$

Algorithm 2 Proxy-based process

Inputs :
- scenario sampler $\mathbb{X}$
- network simulator $\mathcal{S}$
- MGP proxy $\tilde{\mathcal{S}}$
- confidence interval precise enough : $CI_{good}?$ tool
- confidence interval level $1-\alpha$

Output : $I = [p_{min}, p_{max}]$, $1-\alpha$ confidence interval level around $\mathbf{p_{safe}}$

Hyperparameters :
- Batch size m_{batch}
- MGP parameters
- Proxy usage confidence threshold p_{inf} and p_{sup}

```
1:  Initialize $m = 0$ and $I_m = [0, 1]$
2:  Initialize proxy $\tilde{\mathcal{S}}$
3:  while not $CI_{good}?(I_m)$ do
4:      Draw $m_{batch}$ scenarios $\boldsymbol{x_1}, ..., \boldsymbol{x_{m_{batch}}}$ using $\mathbb{X}$
5:      Increase $m = m + 1$
6:      for $i \in [1, m_{batch}]$ do
7:          Compute the probability $p_i = \mathbb{P}(z_i = 1) = \tilde{\mathcal{S}}(\boldsymbol{x_i})$
8:          if $p_i \leq P_{min}$ or $p_i \geq P_{max}$ then
9:              Draw $w_i \sim B(p_i)$
10:             if $w_i = 1$ then
11:                 Set $q_i = p_i$
12:             else
13:                 Set $q_i = 1 - p_i$
14:             end if
15:         else
16:             Set $w_i = \mathcal{S}(\boldsymbol{x_i})$
17:             Set $q_i = 1$
18:         end if
19:         Add outcome $(w_i, q_i)$ to the bag $\mathcal{B} = \mathcal{B} \cup \{(w_i, q_i)\}$
20:     end for
21:     Update MGP parameters with simulated samples in $\mathcal{B}$
22:     Compute $I_m$ of level $1 - \alpha$ using $\mathcal{B}$
23: end while
```

We suppose that we observe $w_1, ..., w_m$ and $q_1, ..., q_m$, we look for a confidence interval of level $1-\alpha$ around $\mathbf{p_{safe}}$.

Proposition 1

We set :

$$\overline{Z_m} = \frac{\sum_{i=1}^m w_i - (1-q_i)}{\sum_{i=1}^m 2q_i - 1}$$
$$v_m = \frac{\left(\sum_{i=1}^m 2q_i - 1\right)^2}{\sum_{i=1}^m (2q_i - 1)^2}$$
$$\sigma_m^2 = \frac{\sum_{i=1}^m q_i(1-q_i)}{\sum_{i=1}^m (2q_i - 1)^2}$$

Then,

$$\sqrt{v_m}\frac{\overline{Z_m} - \mathbf{p_{safe}}}{\sqrt{\sigma_m^2 + \mathbf{p_{safe}}(1-\mathbf{p_{safe}})}} \longrightarrow_d \mathcal{N}(0,1)$$

Proof : See Annex A

The MGP does not provide us directly w_i, nor $\mathbb{P}(w_i = z_i)$, but only $p_i = \mathbb{P}(z_i = 1 | MGP\ prediction)$. The previous proposition cannot be directly applied. The issue is overcome by generating w_i using a Bernoulli distribution $B(p_i)$ and choosing $q_i = p_i$ if $w_i = 1$ or $q_i = 1 - p_i$ if $w_i = 0$. Then, we have $\mathbb{P}(w_i = z_i) = q_i$ in all cases. All the information the MGP provides is contained in q_i, and the random drawings ensure the correctness of the propositions.

If there is no uncertainty on the observations $w_1, ..., w_m$, i.e. $\forall i \in [1, p], q_i = 1$, then $w_i \sim B(\mathbf{p_{safe}})$ and we retrieve the CLT for the z_i. On the contrary, if predictions provide no information, i.e., $\forall i \in [1, p], q_i = 0.5$, then $w_i \sim B(1/2)$ and we retrieve the CLT for balanced Bernoulli variables. Our resulting CLT is a combination of these two CLT. The balance between the two is determined by the q_i. Both terms $\overline{Z_p}$ and v_p can be interpreted as the average $\overline{w}$ and p while σ_p^2 is an additional variance term. It represents a measure of the quantity of uncertainty contained in the observations. The evolution of this term will be studied in the experiments.

Proposition 2

Denoting by Q_α the $1-\alpha$ quantile of the standard normal distribution, the asymptotic confidence interval around $\mathbf{p_{safe}}$ is $[p_{min}, p_{max}]$ with :

$$p_{min} = \frac{\overline{Z_m} + \frac{Q_{\frac{\alpha}{2}}^2}{2v_m} - Q_{\frac{\alpha}{2}}\sqrt{\frac{\overline{Z_m}(1-\overline{Z_m})}{v_m} + \frac{\sigma_m^2}{v_m}\left[1 + \frac{Q_{\frac{\alpha}{2}}^2}{v_m}\right] + \frac{Q_{\frac{\alpha}{2}}^2}{4v_m^2}}}{1 + \frac{Q_{\frac{\alpha}{2}}^2}{v_m}}$$

$$p_{max} = \frac{\overline{Z_m} + \frac{Q_{\frac{\alpha}{2}}^2}{2v_m} + Q_{\frac{\alpha}{2}}\sqrt{\frac{\overline{Z_m}(1-\overline{Z_m})}{v_m} + \frac{\sigma_m^2}{v_m}\left[1 + \frac{Q_{\frac{\alpha}{2}}^2}{v_m}\right] + \frac{Q_{\frac{\alpha}{2}}^2}{4v_m^2}}}{1 + \frac{Q_{\frac{\alpha}{2}}^2}{v_m}}$$

Proof : See Annex B

IV. Computational results

A. Details on the case study

Simulator details

Both processes' performances are tested on a small network of $L = 5$ lines and $N = 10$ generator nodes. Flows on the lines are computed using the PTDF [24], and P^{max} values, considered as features of the zone. When faraway topological changes occur outside the area, it usually results in modifying the PTDF values. Including topological changes in the scenario $\boldsymbol{x}$ would result in an exploding dimension, making the MGP inefficient. Such phenomenon is thus considered a disturbance: at each simulation, we add a small white noise ϵ_n to the PTDF. Our modeling, however, does not embody significant changes

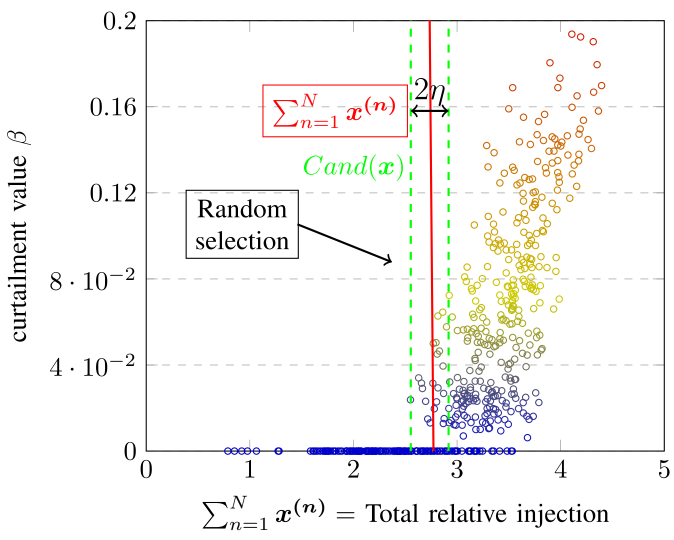


Fig. 4. Selection of the curtailment value β for a given scenario based on historical curtailment decisions

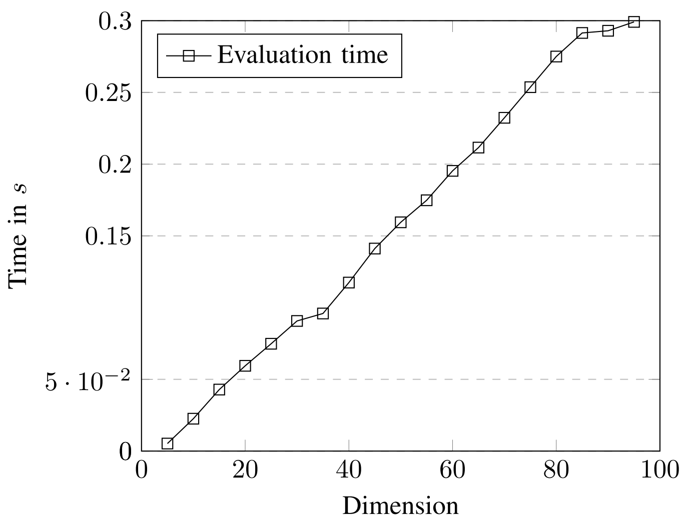


Fig. 5. Integral evaluation time depending on the dimension, 10000 averaged simulations per point

close to the zone, as it would completely redefine the PTDF values. It is one of the limitations of this model.

The real NAZA controller cannot be integrated into our simulator. Our scenarios have no temporal dimension, and NAZA actions are based on the past flow increase estimation. To model fictitious actions, we thus add a $\beta \in [0, 1]$ coefficient that curtails the available renewable power for all nodes.

$$F_l = \beta \sum_{n=1}^{N} (PTDF_{l,n} + \epsilon_n) x_n$$

β is selected based on the minimum required curtailment that avoids congestion for scenarios with similar total power injection in the zone. More specifically, given a set of scenarios $\boldsymbol{x_1}, ..., \boldsymbol{x_J}$, we compute the minimum required curtailment that avoids congestion: $\beta_j = \arg\max_{\beta\in[0,1]} \left\{ \beta \sum_{n=1}^{N} (PTDF_{l,n} + \epsilon_n) x_n^j < 1 \right\}$. Then, for another scenario $\boldsymbol{x}$, we randomly draw its curtailment value β among the set of candidates : $Cand(\boldsymbol{x}) = \left\{ \beta_j, j \in [1, J], \left| \sum_{n=1}^{N} x_n - x_n^j \right| < \eta \right\}$, with default value to 1 if the set is empty. Fig.4 illustrates the β selection procedure. Again, the goal of this simple fictitious simulator is only to illustrate the proxy-based process. It will be tested with real network simulators, including a real NAZA automaton.

Convergence criterion
We sample random scenarios until a certain amount of simulations has been reached. For the brute-force process, the number of simulations matches the number of sampled scenarios. More iterations can be performed for the proxy-based process, depending on the proxy's accuracy.

Computational details Over the iterations, we might have tens of thousands of simulated scenarios to be considered to train the MGP. It, however, implies inverting a covariance matrix with a dimension equal to the number of simulated scenarios, thus over tens of thousands. It is too time and space-consuming to be efficiently computed. Nevertheless, when computing the likelihood of a new scenario, many previously simulated scenarios will be very different. Their correlation with the new input will be negligible. All these scenarios could thus be neglected when estimating the parameters of the conditional distribution. Inspired by [21], we consider only a small proportion of the simulated scenarios when computing the likelihood of a new input $\boldsymbol{x_0}$.

- At most N^* scenarios
- Only neighbors scenarios that are at a distance less than d^*: $d(\boldsymbol{x}, \boldsymbol{x_0}) < d^*$

The last point that deserves to be highlighted is the computation of the integral for p_i. Both processes are implemented in Python; we use the Scipy library [22] to approximate the integral numerically. In small dimensions, the evaluation of the integral is fast. In higher dimensions, the time quickly increases, cf Fig.5. It is another limitation to overcome to generalize this approach to bigger networks.

B. *Proxy performances*

In this subsection, we illustrate how the MGP manages to learn the behavior of the black-box function to act as a proxy of the simulator. We plot three different indicators.

First, we display in Fig.6 the evolution of σ_* across the process's iterations. σ_* is the scale of the covariance matrix of the posterior distribution. A small σ_* reflects a low variance and thus a high confidence in the predicted outcome. σ_* entirely depends on the point we wish to predict, rendering the plot irregular. To catch the global trend, we display a moving average.

In the beginning, the MGP has no observations to learn from. The variance of the posterior distribution is $\sigma_* = \sigma_f + \sigma_0 = 1$, an arbitrary prior choice. Throughout the

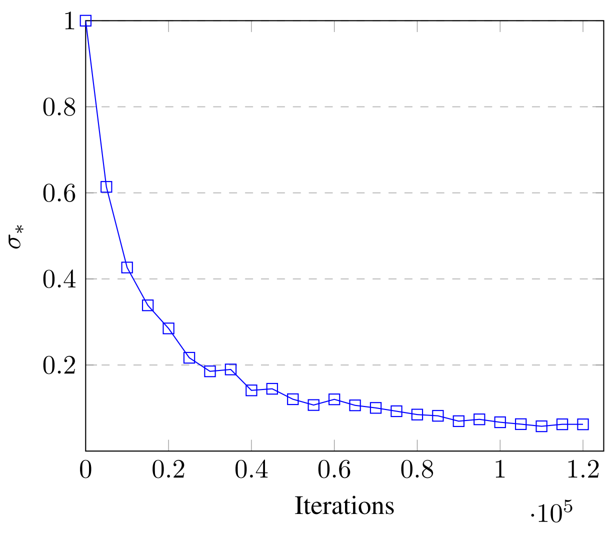


Fig. 6. Evolution of the 200 mobile average of σ_* during the proxy-based process

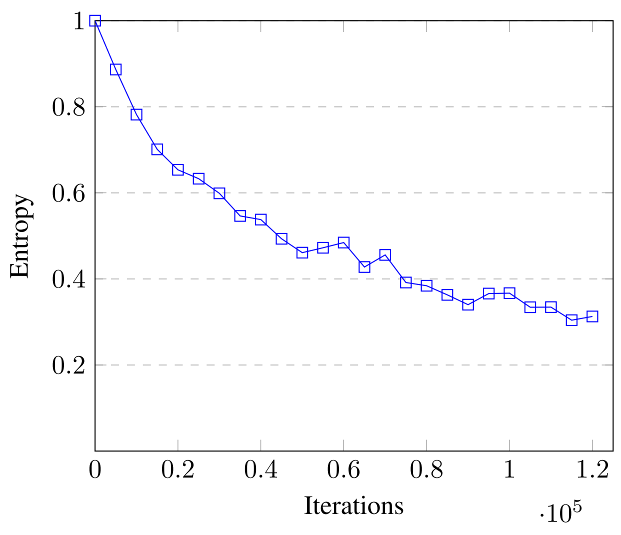


Fig. 7. Evolution of the 200 mobile average of the proxy's prediction entropy during the proxy-based process

process, we simulate more scenarios, increasing the chances of having many correlated points close to the new sampled observation. Mathematically, this boils down to a higher value for the Mahalanobis term $\mathbf{\Sigma_{[1,m],m+1}}^T \mathbf{\Sigma_m}^{-1} \mathbf{\Sigma_{[1,m],m+1}}^T$, and thus a lower variance. The decrease of σ_* illustrates the increasing density of simulated scenarios in the information structure, directly linked to a downward variance posterior distribution and a high confidence in the prediction. σ_* could amount to a loss function, whose decrease attests that our MGP indeed learns the simulator's answer.

The second indicator is related to the evolution of confidence in the outcome's prediction across the process's iterations. We plot in Fig.7 the evolution of the prediction's entropy: $H(z_i) = \frac{-p_i log(p_i) - (1-p_i) log(1-p_i)}{log(2)}$. A small entropy characterizes a probability close to 0 or 1, thus a confident prediction, while a higher entropy stands for a more uncertain prediction. Again, we display a moving average to avoid a very irregular plot.

In the beginning, the posterior distribution is $\mathcal{N}(0, \mathbf{\Omega})$. $\mathbf{\Omega}$ scale is adapted so that the prior probability is unbiased: $p_1 = q_1 = \frac{1}{2}$ and $H(z_1) = 1$. The congestion probability is mainly influenced by whether or not μ_* is close, relatively to σ_*, to the congestion limit, i.e., the boundaries of the $[0,1]^N$ volume. As σ_* decreases, it becomes less and less likely to be close to the border. Thus, the computed probabilities are increasingly extreme, close to 0 (confidence in an overload) or very close to 1 (confidence in a safe line).

Finally, the last interesting factor to display is the % of simulated scenarios since the beginning of the certification process. The lower the %, is, the more simulations were avoided, and thus the better the proxy is.

Initially, the proxy's confidence is insufficient to replace a real simulation, and most scenarios are simulated. Then, as the proxy quality improves and the prediction's entropy decreases, the proxy is used more and more often. Over the last 1000 of sampled scenarios, less than 50% are simulated, leading to 20% of avoided simulations.

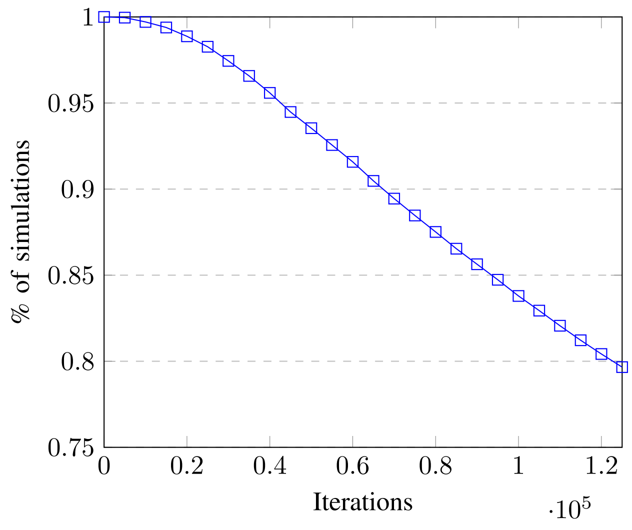


Fig. 8. Evolution of the total % of simulated scenarios since the beginning during the proxy-based process

C. *Processes performances*

To conclude the experiments section, we present the performances of each process for probability estimation. Given a maximum amount of simulations, we compare in Table 1 the estimated $\mathbf{p_{safe}}$ probability and the confidence interval obtained by both processes.

The brute-force process performs 100,000 iterations and reaches a relative error of 0.05%. Comparatively, the proxy-based process gets over 125,000 iterations and avoids more than 25,000 simulations. The proxy-based confidence interval is computed with more iterations, leading to a smaller relative error and a smaller length. The performance gap is almost 20% above the brute-force process. Since the proxy is con-

$\mathbf{p_{safe}} = 0.9044$		
	Brute-force	Proxy-based
Number of iterations	125575	100000
p_{min}	0.9018	0.9023
p_{max}	0.9067	0.9064
Relative error	0.05%	0.041%
Confidence interval length	0.0049	0.0041

TABLE I
PROCESSES PERFORMANCES COMPARISON

tinuously enhanced across the simulations, the gap between the two processes would be even more significant for a higher maximum amount of simulations.

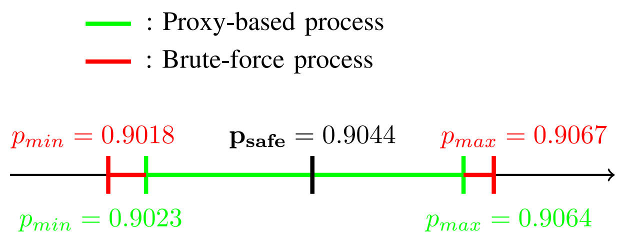


Fig. 9. Confidence intervals obtained by both processes

V. Conclusion

In this work, we proposed two processes to estimate the probability of observing congestion on a network. The network and its simulator are a black-box function that outputs the maximum flow observed on each line for a given renewable energy production scenario. The brute-force process draws and simulates scenarios until it reaches a satisfying probability estimation. As power system simulations are often very long, it is time-consuming. The proxy-based process trains a Multivariate Gaussian Process to predict the black box's answer, with an exact confidence interval on the prediction. The simulation is avoided when the proxy is confident enough in the prediction. For a given budget of simulations, it leads to a much more accurate probability estimation, with an error reduced by 20%. Many simulations are not performed, allowing the process to run more iterations. The MGP manages here to efficiently learn the multivariate answer of a black-box function with correlated components while providing explicability and theoretical guarantees about the prediction's error. Multivariate Gaussian processes are of great interest for approximating power systems behaviors and can be applied to many problems.

We supposed in this work that the output's correlation matrix Ω was known, while in practice, it has to be learned. A first extension of this work will be to propose an estimation method of this matrix during the process. Also, the kernel choice can be discussed. This paper aimed to illustrate the proxy-based certification process on a simple black box involving low-dimensional inputs. We aim to extend it to more realistic simulations with a real network and complex scenarios.

References

[1] Hoang, Duc-Trung and Olaru, Sorin and Iovine, Alessio and Maeght, Jean and Panciatici, Patrick and Ruiz, Manuel, *Power Congestion Management of a sub-Transmission Area Power Network using Partial Renewable Power Curtailment via MPC*, 2021 60th IEEE Conference on Decision and Control (CDC)

[2] Hoang, Duc-Trung and Olaru, Sorin and Iovine, Alessio and Maeght, Jean and Panciatici, Patrick and Ruiz, Manuel, *Predictive Control for Zonal Congestion Management of a Transmission Network*, 2021 29th Mediterranean Conference on Control and Automation (MED)

[3] Straub, Clementine and Olaru, Sorin and Maeght, Jean and Panciatici, Patrick, *Zonal Congestion Management Mixing Large Battery Storage Systems and Generation Curtailment*, 2018 IEEE Conference on Control Technology and Applications (CCTA)

[4] Meyer, Bruno and Astic, Jean-Yves and Meyer, Pierre and Sardou, François-Xavier and Poumarede, Christian and Couturier, Nicolas and Fontaine, Mathieu and Lemaitre, Christian and Maeght, Jean and Straub, Clémentine, *Power Transmission Technologies and Solutions: The Latest Advances at RTE, the French Transmission System Operator*, IEEE Power and Energy Magazine

[5] Fischer, Hans, *A history of the central limit theorem: from classical to modern probability theory*, 2011 Springer

[6] Jeffrey T. Morisette and Siamak Khorram, *Exact Binomial Confidence Interval for Proportions*, 1998 PHOTOGRAMMETRIC ENGINEERING & REMOTE SENSING

[7] Canyasse, Raphaël and Dalal, Gal and Mannor, Shie, *Supervised learning for optimal power flow as a real-time proxy* 2017 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)

[8] Duchesne, Laurine and Karangelos, Efthymios and Wehenkel, Louis, *Using Machine Learning to Enable Probabilistic Reliability Assessment in Operation Planning*, 2018 Power Systems Computation Conference (PSCC)

[9] Dalal, Gal and Gilboa, Elad and Mannor, Shie and Wehenkel, Louis, *Unit Commitment Using Nearest Neighbor as a Short-Term Proxy* 2018 Power Systems Computation Conference (PSCC)

[10] Dalal, Gal and Gilboa, Elad and Mannor, Shie and Wehenkel, Louis, *Chance-Constrained Outage Scheduling Using a Machine Learning Proxy* 2019 IEEE Transactions on Power Systems

[11] Wenbo Chen and Seonho Park and Mathieu Tanneau and Pascal Van Hentenryck, *Learning optimization proxies for large-scale Security-Constrained Economic Dispatch*, 2022 Electric Power Systems Research

[12] Rasmussen, Bousquet, O., von Luxburg, U., Rätsch, G, *Gaussian Processes in Machine Learning*, Advanced Lectures on Machine Learning. ML 2003. Lecture Notes in Computer Science, vol 3176. Springer, Berlin, Heidelberg

[13] Kocijan, Jus, *Modelling and Control of Dynamic Systems Using Gaussian Process Models*, 2016 Springer

[14] Jain, Achin and Nghiem, Truong and Morari, Manfred and Mangharam, Rahul, *Learning and Control Using Gaussian Processes* 2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)

[15] Yandong Yang and Shufang Li and Wenqi Li and Meijun Qu, *Power load probability density forecasting using Gaussian process quantile regression* 2013 Applied Energy

[16] Ana M. Ospina and Yue Chen and Andrey Bernstein and Emiliano Dall'Anese, *Learning-based demand response in grid-interactive buildings via Gaussian Processes*, 2011 Electric Power Systems Research

[17] Zexun Chen and Bo Wang and Alexander N. Gorban, *Multivariate Gaussian and Student-t process regression for multi-output prediction*, 2019 Springer Science and Business Media LLC

[18] Duvenaud David, *Automatic model construction with Gaussian processes*, 2014 PhD Thesis

[19] Mathai Arak and Provost Serge and Haubold Hans, *Multivariate Statistical Analysis in the Real and Complex Domains, Chapter 4: The Matrix-Variate Gaussian Distribution*, 2022 Springer International Publishing, Pages 217-288

[20] Marin, Jose Luis and Gaitan, Vicenç and Oms, Guiu and Chiaramello, Marco and Cossart, Quentin and Guironnet, Adrien, *An Open Source Tool to Compare Simulators on Large-Scale Cases — Application to Dynawo*, 2022 Open Source Modelling and Simulation of Energy Systems (OSMSES)

[21] Liu, Haitao and Ong, Yew-Soon and Shen, Xiaobo and Cai, Jianfei, *When Gaussian Process Meets Big Data: A Review of Scalable GPs*, 2020 IEEE Transactions on Neural Networks and Learning Systems

[22] Virtanen, Pauli and Gommers, Ralf and Oliphant, Travis E. and Haberland, Matt and Reddy, Tyler and Cournapeau, David and Burovski, Evgeni and Peterson, Pearu and Weckesser, Warren and Bright, Jonathan and van der Walt, Stéfan J. and Brett, Matthew and Wilson, Joshua and Millman, K. Jarrod and Mayorov, Nikolay and Nelson, Andrew R. J. and Jones, Eric and Kern, Robert and Larson, Eric and Carey, C J and Polat, İlhan and Feng, Yu and Moore, Eric W. and VanderPlas, Jake and Laxalde, Denis and Perktold, Josef and Cimrman, Robert and Henriksen, Ian and Quintero, E. A. and Harris, Charles R. and Archibald, Anne M. and Ribeiro, Antônio H. and Pedregosa, Fabian and van Mulbregt, Paul and SciPy 1.0 Contributors, *SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python*, 2020 Nature Methods

[23] Nowak, Piotr and Hryniewicz, Olgierd, *Generalized versions of MV-algebraic central limit theorems*, 2015 Kybernetika

[24] Šošić, Darko and Škokljev, Ivan and Pokimica, Nemanja, *Features of Power Transfer Distribution Coefficients in power System Networks*, 2014

ANNEX A: PROOF OF PROPOSITION 1

The proof relies on the Lyapunov version of the CLT [23]. It states that for m given random independent variables $X_1, .., X_m$, with expected value μ_i and variance σ_i^2, if for some $\delta > 0$, Lyapunov condition is satisfied :

$$s_m^2 = \sum_{i=1}^{m} \sigma_i^2$$

$$\lim_{m\to\infty} \frac{1}{s_m^2} \sum_{i=1}^{m} \mathbb{E}\left[|X_i - \mu_i|^{2+\delta}\right] = 0$$

Then,

$$\frac{1}{s_m} \sum_{i=1}^{m} (X_i - \mu_i) \longrightarrow_d \mathcal{N}(0,1)$$

In our case, one can easily check that $\forall i \in [1, m]$:

$$\begin{aligned} w_i &\sim B\left((2q_i - 1)\mathbf{p_{safe}} + (1 - q_i)\right) \\ \mathbb{E}\left[w_i\right] &= (2q_i - 1)\mathbf{p_{safe}} + (1 - q_i) \\ V(w_i) &= q_i(1 - q_i) + \mathbf{p_{safe}}(1 - \mathbf{p_{safe}})(2q_i - 1)^2 \end{aligned}$$

Let $A_1, ..., A_m$ be m random independant variables following the Bernoulli distributions $B(a_1), ..., B(a_m)$, then :

$$\begin{aligned} s_m^2 &= \sum_{i=1}^{m} a_i(1 - a_i) \\ \mathbb{E}\left[|A_i - \mu_i|^{2+\delta}\right] &= a_i(1 - a_i)^{2+\delta} + (1 - a_i)a_i^{2+\delta} \\ &\leq 2a_i(1 - a_i) \end{aligned}$$

Since we suppose that the simulator's answer is non-deterministic, there is always a residual uncertainty on the result's prediction. It implies that $q_i(1 - q_i)$ does not converge toward 0 and thus the serie $\sum_{i=1}^{m} q_i(1 - q_i)$ diverges toward $+\infty$. We deduce that :

$$\begin{aligned} \lim_{m\to\infty} \sum_{i=1}^{m} \frac{\mathbb{E}\left[|A_i - \mu_i|^{2+\delta}\right]}{s_p^{2+\delta}} &\leq \lim_{m\to\infty} \frac{2\sum_{i=1}^{m} a_i(1 - a_i)}{\left(\sum_{i=1}^{m} a_i(1 - a_i)\right)^{1+\frac{\delta}{2}}} \\ &= 0 \end{aligned}$$

Lyapunov's condition is thus verified for our set of Bernoulli variables. We apply the theorem to $W_1, ..., W_m$:

$$\begin{aligned} &\frac{\sum_{i=1}^{m} W_i - (2q_i - 1)\mathbf{p_{safe}} - (1 - q_i)}{\sqrt{\sum_{i=1}^{m} q_i(1 - q_i) + \mathbf{p_{safe}}(1 - \mathbf{p_{safe}})(2q_i - 1)^2}} \\ =&\frac{\sum_{i=1}^{m} W_i - (1 - q_i) - \mathbf{p_{safe}} \sum_{i=1}^{m}(2q_i - 1)}{\sqrt{\sum_{i=1}^{m} q_i(1 - q_i) + \mathbf{p_{safe}}(1 - \mathbf{p_{safe}}) \sum_{i=1}^{m}(2q_i - 1)^2}} \\ =&\sqrt{\frac{\left(\sum_{i=1}^{m}(2q_i - 1)\right)^2}{\sum_{i=1}^{m}(2q_i - 1)^2}} \frac{\frac{\sum_{i=1}^{m} W_i - (1-q_i)}{\sum_{i=1}^{m}(2q_i - 1)} - \mathbf{p_{safe}}}{\sqrt{\frac{\sum_{i=1}^{m} q_i(1-q_i)}{\sum_{i=1}^{m}(2q_i-1)^2} + \mathbf{p_{safe}}(1 - \mathbf{p_{safe}})}} \\ =&\sqrt{v_m} \frac{\overline{Z_m} - \mathbf{p_{safe}}}{\sqrt{\sigma_m^2 + \mathbf{p_{safe}}(1 - \mathbf{p_{safe}})}} \end{aligned}$$

ANNEX B: PROOF OF PROPOSITION 2

Asymptotically, we know that $\sqrt{v_m}\frac{\overline{Z_m} - \mathbf{p_{safe}}}{\sqrt{\sigma_m^2 + \mathbf{p_{safe}}(1 - \mathbf{p_{safe}})}} \sim \mathcal{N}(0, 1)$. To obtain the result, we solve the following inequality :

$$\begin{aligned} &- q_{\frac{\alpha}{2}} \leq \sqrt{v_m} \frac{\overline{Z_m} - \mathbf{p_{safe}}}{\sqrt{\sigma_m^2 + \mathbf{p_{safe}}(1 - \mathbf{p_{safe}})}} \leq q_{\frac{\alpha}{2}} \\ &\implies v_m \frac{\left(\overline{Z_m} - \mathbf{p_{safe}}\right)^2}{\sigma_m^2 + \mathbf{p_{safe}}(1 - \mathbf{p_{safe}})} \leq q_{\frac{\alpha}{2}}^2 \\ &\implies \overline{Z_m}^2 + \mathbf{p_{safe}}^2 - 2\mathbf{p_{safe}}\overline{Z_m} \leq \sigma_m^2 \frac{q_{\frac{\alpha}{2}}^2}{v_m} + \mathbf{p_{safe}} \frac{q_{\frac{\alpha}{2}}^2}{v_m} \\ &- \mathbf{p_{safe}}^2 \frac{q_{\frac{\alpha}{2}}^2}{v_m} \\ &\implies \left[1 + \frac{q_{\frac{\alpha}{2}}^2}{v_m}\right] \mathbf{p_{safe}}^2 + \left[-2\overline{Z_m} - \frac{q_{\frac{\alpha}{2}}^2}{v_m}\right] \mathbf{p_{safe}} \\ &+ \left[\overline{Z_m}^2 - \sigma_m^2 \frac{q_{\frac{\alpha}{2}}^2}{v_m}\right] \leq 0 \end{aligned}$$

$$\begin{aligned} \Delta &= \left[2\overline{Z_m} + \frac{q_{\frac{\alpha}{2}}^2}{v_m}\right]^2 - 4\left[1 + \frac{q_{\frac{\alpha}{2}}^2}{v_m}\right]\left[\overline{Z_m}^2 - \sigma_m^2 \frac{q_{\frac{\alpha}{2}}^2}{v_m}\right] \\ &= 4\overline{Z_m}^2 + \frac{q_{\frac{\alpha}{2}}^4}{v_m^2} + 4\overline{Z_m}\frac{q_{\frac{\alpha}{2}}^2}{v_m} - 4\overline{Z_m}^2 + 4\sigma_m^2 \frac{q_{\frac{\alpha}{2}}^2}{v_m} \\ &- 4\overline{Z_m}^2 \frac{q_{\frac{\alpha}{2}}^2}{v_m} + 4\sigma_m^2 \frac{q_{\frac{\alpha}{2}}^4}{v_m^2} \\ &= 4\frac{q_{\frac{\alpha}{2}}^2}{v_m}\left[\overline{Z_m}\left(1 - \overline{Z_m}\right) + \sigma_m^2\left[1 + \frac{q_{\frac{\alpha}{2}}^2}{v_m}\right] + \frac{q_{\frac{\alpha}{2}}^2}{4v_m}\right] \end{aligned}$$

We can finally derive the expressions of p_{min} and p_{max} :

$$p_{min} = \frac{\overline{Z_m} + \frac{q_{\frac{\alpha}{2}}^2}{2v_m} - q_{\frac{\alpha}{2}} \sqrt{\frac{\overline{Z_m}\left(1-\overline{Z_m}\right)}{v_m} + \frac{\sigma_m^2}{v_m}\left[1 + \frac{q_{\frac{\alpha}{2}}^2}{v_m}\right] + \frac{q_{\frac{\alpha}{2}}^2}{4v_m^2}}}{1 + \frac{q_{\frac{\alpha}{2}}^2}{v_m}}$$

$$p_{max} = \frac{\overline{Z_m} + \frac{q_{\frac{\alpha}{2}}^2}{2v_m} + q_{\frac{\alpha}{2}} \sqrt{\frac{\overline{Z_m}\left(1-\overline{Z_m}\right)}{v_m} + \frac{\sigma_m^2}{v_m}\left[1 + \frac{q_{\frac{\alpha}{2}}^2}{v_m}\right] + \frac{q_{\frac{\alpha}{2}}^2}{4v_m^2}}}{1 + \frac{q_{\frac{\alpha}{2}}^2}{v_m}}$$